RSA-1

November 10, 2025

[1]:
```python
# Eratostenove sito na hladanie prvocisel mensich ako zadane n
# from math import sqrt

def getPrimes(n):
    sito = [1 for k in range(0,n)]
    # for i in range(2,int(sqrt(n+1))):
    for i in range(2,n):
        if sito[i] == 1:
            for j in range(2*i,n,i):
                sito[j] = 0

    primes = [k for k in range(2,n) if sito[k] == 1]
    return primes
```

[2]:
```python
# vytvorenie zoznamu prvocisel

primes = getPrimes(10000)
print("Zoznam prvocisel:", primes)
```

```
Zoznam prvocisel: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53,
59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139,
149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229,
233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317,
331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421,
431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521,
523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619,
631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733,
739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839,
853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947, 953,
967, 971, 977, 983, 991, 997, 1009, 1013, 1019, 1021, 1031, 1033, 1039, 1049,
1051, 1061, 1063, 1069, 1087, 1091, 1093, 1097, 1103, 1109, 1117, 1123, 1129,
1151, 1153, 1163, 1171, 1181, 1187, 1193, 1201, 1213, 1217, 1223, 1229, 1231,
1237, 1249, 1259, 1277, 1279, 1283, 1289, 1291, 1297, 1301, 1303, 1307, 1319,
1321, 1327, 1361, 1367, 1373, 1381, 1399, 1409, 1423, 1427, 1429, 1433, 1439,
1447, 1451, 1453, 1459, 1471, 1481, 1483, 1487, 1489, 1493, 1499, 1511, 1523,
1531, 1543, 1549, 1553, 1559, 1567, 1571, 1579, 1583, 1597, 1601, 1607, 1609,
1613, 1619, 1621, 1627, 1637, 1657, 1663, 1667, 1669, 1693, 1697, 1699, 1709,
1721, 1723, 1733, 1741, 1747, 1753, 1759, 1777, 1783, 1787, 1789, 1801, 1811,
```

1823, 1831, 1847, 1861, 1867, 1871, 1873, 1877, 1879, 1889, 1901, 1907, 1913,
1931, 1933, 1949, 1951, 1973, 1979, 1987, 1993, 1997, 1999, 2003, 2011, 2017,
2027, 2029, 2039, 2053, 2063, 2069, 2081, 2083, 2087, 2089, 2099, 2111, 2113,
2129, 2131, 2137, 2141, 2143, 2153, 2161, 2179, 2203, 2207, 2213, 2221, 2237,
2239, 2243, 2251, 2267, 2269, 2273, 2281, 2287, 2293, 2297, 2309, 2311, 2333,
2339, 2341, 2347, 2351, 2357, 2371, 2377, 2381, 2383, 2389, 2393, 2399, 2411,
2417, 2423, 2437, 2441, 2447, 2459, 2467, 2473, 2477, 2503, 2521, 2531, 2539,
2543, 2549, 2551, 2557, 2579, 2591, 2593, 2609, 2617, 2621, 2633, 2647, 2657,
2659, 2663, 2671, 2677, 2683, 2687, 2689, 2693, 2699, 2707, 2711, 2713, 2719,
2729, 2731, 2741, 2749, 2753, 2767, 2777, 2789, 2791, 2797, 2801, 2803, 2819,
2833, 2837, 2843, 2851, 2857, 2861, 2879, 2887, 2897, 2903, 2909, 2917, 2927,
2939, 2953, 2957, 2963, 2969, 2971, 2999, 3001, 3011, 3019, 3023, 3037, 3041,
3049, 3061, 3067, 3079, 3083, 3089, 3109, 3119, 3121, 3137, 3163, 3167, 3169,
3181, 3187, 3191, 3203, 3209, 3217, 3221, 3229, 3251, 3253, 3257, 3259, 3271,
3299, 3301, 3307, 3313, 3319, 3323, 3329, 3331, 3343, 3347, 3359, 3361, 3371,
3373, 3389, 3391, 3407, 3413, 3433, 3449, 3457, 3461, 3463, 3467, 3469, 3491,
3499, 3511, 3517, 3527, 3529, 3533, 3539, 3541, 3547, 3557, 3559, 3571, 3581,
3583, 3593, 3607, 3613, 3617, 3623, 3631, 3637, 3643, 3659, 3671, 3673, 3677,
3691, 3697, 3701, 3709, 3719, 3727, 3733, 3739, 3761, 3767, 3769, 3779, 3793,
3797, 3803, 3821, 3823, 3833, 3847, 3851, 3853, 3863, 3877, 3881, 3889, 3907,
3911, 3917, 3919, 3923, 3929, 3931, 3943, 3947, 3967, 3989, 4001, 4003, 4007,
4013, 4019, 4021, 4027, 4049, 4051, 4057, 4073, 4079, 4091, 4093, 4099, 4111,
4127, 4129, 4133, 4139, 4153, 4157, 4159, 4177, 4201, 4211, 4217, 4219, 4229,
4231, 4241, 4243, 4253, 4259, 4261, 4271, 4273, 4283, 4289, 4297, 4327, 4337,
4339, 4349, 4357, 4363, 4373, 4391, 4397, 4409, 4421, 4423, 4441, 4447, 4451,
4457, 4463, 4481, 4483, 4493, 4507, 4513, 4517, 4519, 4523, 4547, 4549, 4561,
4567, 4583, 4591, 4597, 4603, 4621, 4637, 4639, 4643, 4649, 4651, 4657, 4663,
4673, 4679, 4691, 4703, 4721, 4723, 4729, 4733, 4751, 4759, 4783, 4787, 4789,
4793, 4799, 4801, 4813, 4817, 4831, 4861, 4871, 4877, 4889, 4903, 4909, 4919,
4931, 4933, 4937, 4943, 4951, 4957, 4967, 4969, 4973, 4987, 4993, 4999, 5003,
5009, 5011, 5021, 5023, 5039, 5051, 5059, 5077, 5081, 5087, 5099, 5101, 5107,
5113, 5119, 5147, 5153, 5167, 5171, 5179, 5189, 5197, 5209, 5227, 5231, 5233,
5237, 5261, 5273, 5279, 5281, 5297, 5303, 5309, 5323, 5333, 5347, 5351, 5381,
5387, 5393, 5399, 5407, 5413, 5417, 5419, 5431, 5437, 5441, 5443, 5449, 5471,
5477, 5479, 5483, 5501, 5503, 5507, 5519, 5521, 5527, 5531, 5557, 5563, 5569,
5573, 5581, 5591, 5623, 5639, 5641, 5647, 5651, 5653, 5657, 5659, 5669, 5683,
5689, 5693, 5701, 5711, 5717, 5737, 5741, 5743, 5749, 5779, 5783, 5791, 5801,
5807, 5813, 5821, 5827, 5839, 5843, 5849, 5851, 5857, 5861, 5867, 5869, 5879,
5881, 5897, 5903, 5923, 5927, 5939, 5953, 5981, 5987, 6007, 6011, 6029, 6037,
6043, 6047, 6053, 6067, 6073, 6079, 6089, 6091, 6101, 6113, 6121, 6131, 6133,
6143, 6151, 6163, 6173, 6197, 6199, 6203, 6211, 6217, 6221, 6229, 6247, 6257,
6263, 6269, 6271, 6277, 6287, 6299, 6301, 6311, 6317, 6323, 6329, 6337, 6343,
6353, 6359, 6361, 6367, 6373, 6379, 6389, 6397, 6421, 6427, 6449, 6451, 6469,
6473, 6481, 6491, 6521, 6529, 6547, 6551, 6553, 6563, 6569, 6571, 6577, 6581,
6599, 6607, 6619, 6637, 6653, 6659, 6661, 6673, 6679, 6689, 6691, 6701, 6703,
6709, 6719, 6733, 6737, 6761, 6763, 6779, 6781, 6791, 6793, 6803, 6823, 6827,
6829, 6833, 6841, 6857, 6863, 6869, 6871, 6883, 6899, 6907, 6911, 6917, 6947,
6949, 6959, 6961, 6967, 6971, 6977, 6983, 6991, 6997, 7001, 7013, 7019, 7027,

```
7039, 7043, 7057, 7069, 7079, 7103, 7109, 7121, 7127, 7129, 7151, 7159, 7177,
7187, 7193, 7207, 7211, 7213, 7219, 7229, 7237, 7243, 7247, 7253, 7283, 7297,
7307, 7309, 7321, 7331, 7333, 7349, 7351, 7369, 7393, 7411, 7417, 7433, 7451,
7457, 7459, 7477, 7481, 7487, 7489, 7499, 7507, 7517, 7523, 7529, 7537, 7541,
7547, 7549, 7559, 7561, 7573, 7577, 7583, 7589, 7591, 7603, 7607, 7621, 7639,
7643, 7649, 7669, 7673, 7681, 7687, 7691, 7699, 7703, 7717, 7723, 7727, 7741,
7753, 7757, 7759, 7789, 7793, 7817, 7823, 7829, 7841, 7853, 7867, 7873, 7877,
7879, 7883, 7901, 7907, 7919, 7927, 7933, 7937, 7949, 7951, 7963, 7993, 8009,
8011, 8017, 8039, 8053, 8059, 8069, 8081, 8087, 8089, 8093, 8101, 8111, 8117,
8123, 8147, 8161, 8167, 8171, 8179, 8191, 8209, 8219, 8221, 8231, 8233, 8237,
8243, 8263, 8269, 8273, 8287, 8291, 8293, 8297, 8311, 8317, 8329, 8353, 8363,
8369, 8377, 8387, 8389, 8419, 8423, 8429, 8431, 8443, 8447, 8461, 8467, 8501,
8513, 8521, 8527, 8537, 8539, 8543, 8563, 8573, 8581, 8597, 8599, 8609, 8623,
8627, 8629, 8641, 8647, 8663, 8669, 8677, 8681, 8689, 8693, 8699, 8707, 8713,
8719, 8731, 8737, 8741, 8747, 8753, 8761, 8779, 8783, 8803, 8807, 8819, 8821,
8831, 8837, 8839, 8849, 8861, 8863, 8867, 8887, 8893, 8923, 8929, 8933, 8941,
8951, 8963, 8969, 8971, 8999, 9001, 9007, 9011, 9013, 9029, 9041, 9043, 9049,
9059, 9067, 9091, 9103, 9109, 9127, 9133, 9137, 9151, 9157, 9161, 9173, 9181,
9187, 9199, 9203, 9209, 9221, 9227, 9239, 9241, 9257, 9277, 9281, 9283, 9293,
9311, 9319, 9323, 9337, 9341, 9343, 9349, 9371, 9377, 9391, 9397, 9403, 9413,
9419, 9421, 9431, 9433, 9437, 9439, 9461, 9463, 9467, 9473, 9479, 9491, 9497,
9511, 9521, 9533, 9539, 9547, 9551, 9587, 9601, 9613, 9619, 9623, 9629, 9631,
9643, 9649, 9661, 9677, 9679, 9689, 9697, 9719, 9721, 9733, 9739, 9743, 9749,
9767, 9769, 9781, 9787, 9791, 9803, 9811, 9817, 9829, 9833, 9839, 9851, 9857,
9859, 9871, 9883, 9887, 9901, 9907, 9923, 9929, 9931, 9941, 9949, 9967, 9973]
```

[3]:
```python
from math import sqrt, ceil

def primeTest(n):

    ub = 1 + int(sqrt(n))
    for d in range(2, ub):
        if n % d == 0:
            print("{} deli {}, t.j. {} nie je prvocislo!".format(d, n, n))
            return False

    print("Nenasiel som ziadneho delitela {}, t.j. {} je prvocislo.".format(n,␣
↪n))
    return True
```

[4]:
```python
#primeTest(563419580815451997 83)
primeTest(563)
primeTest(565)
```

```
Nenasiel som ziadneho delitela 563, t.j. 563 je prvocislo.
5 deli 565, t.j. 565 nie je prvocislo!
```

[4]: False

[5]:
```
p = 563
```

[6]:
```
print(pow(135,p-1,p))
print(pow(135,p-2,p))
print(135*367%p)
```

```
1
367
1
```

[7]:
```
import random

# Fermat's test
def fermatTest(p):
    for i in range(5):
        c = random.randrange(2,p-1)
        y = pow(c, p-1, p)
        if (y != 1):
            return False

    return True
```

[8]:
```
fermatTest(56341958081545199783)
```

[8]: False

[9]:
```
carmichael = [561, 1105, 1729, 2465, 2821, 6601, 8911]
for p in carmichael:
    print(p, fermatTest(p))
```

```
561 False
1105 False
1729 True
2465 False
2821 True
6601 False
8911 False
```

[10]:
```
p = 8911
for a in range(2,40):
    print(a, pow(a,p-1,p))
```

```
2 1
3 1
4 1
5 1
6 1
```

```
7 1274
8 1
9 1
10 1
11 1
12 1
13 1
14 1274
15 1
16 1
17 1
18 1
19 7505
20 1
21 1274
22 1
23 1
24 1
25 1
26 1
27 1
28 1274
29 1
30 1
31 1
32 1
33 1
34 1
35 1274
36 1
37 1
38 7505
39 1
```

[ ]:

[11]:
```python
# MILLER - RABIN  primality test
def millRabTest(p, t = 20):
    s, r = 0, p - 1
    while ((r % 2) == 0):
        s, r = s + 1, r // 2

    for i in range(t):

        a = random.randrange(2,p-1)
        y = pow(a, r, p)
```

```
        if (y == 1) or (y == (p-1)):
            continue

        for j in range(1,s):
            y = pow(y, 2, p)
            if (y == (p-1)):
                break
            elif (y == 1):
                break

        if (y != (p-1)):
            return False

    return True
```

[12]:
```
for p in (561, 491):
    print(p, millRabTest(p))
```

```
561 False
491 True
```

[13]:
```
# generovanie nahodneho prvocisla zo zoznamu

import random

def randPrime(primes):
    index = random.randrange(len(primes))
    return primes[index]
```

[14]:
```
# vygenerovanie dvoch roznych prvocisel pre RSA algoritmus

while True:
    p = randPrime(primes)
    q = randPrime(primes)
    if p != q and p > 20 and q > 20:
        break
```

[15]:
```
#p = 605742134588197
#q = 848976880459061

print("Vygenerovane nahodne cisla p =", p,", q =", q)
```

```
Vygenerovane nahodne cisla p = 8929 , q = 3533
```

[16]:
```
# modulus je sucin prvocisel
```

```
n = p * q
print("Modulus n = p * q =", n)
```

```
Modulus n = p * q = 31546157
```

[17]:
```
# vypocet hodnoty Eulerovej funkcie fi(n) pre modulus n
# (pocet nesudelitelnych cisel s n mensich ako n)
# v tomto pripade je to (p-1) * (q - 1)

fi_n = (p - 1) * (q - 1)
print(fi_n)
```

```
31533696
```

[18]:
```
# plati, ze umocnenie lubovolneho x nesudelitelneho s n na fi(n) modulo n musi␣
 ↪byt rovne 1

x = random.randrange(1,n)
#x = 2 * q
exponent = 10 * fi_n + 1
print("x", "=", x)
print(x, "^", exponent, "mod", n, "=", pow(x,exponent,n))
```

```
x = 22387779
22387779 ^ 315336961 mod 31546157 = 22387779
```

[19]:
```
# Euklidov algoritmus na najdenie najvacsieho spolocneho delitela dvoch cisel

def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a
```

[20]:
```
print("Najvacsi spolocny delitel 45 a 61 je", gcd(45, 61))
```

```
Najvacsi spolocny delitel 45 a 61 je 1
```

[21]:
```
# musim zvolit take e, aby bolo nesudelitelne s fi(n)
# t.j. gcd(e,fi(n)) musi byt rovne 1
# e sa voli vacsinou ako relativne male prvocislo (realne som nevidel ine ako␣
 ↪65537)
# nazyva sa aj verejny exponent

for p in primes:
    e = p
    if gcd(e,fi_n) == 1:
        break

print("Verejny exponent e = ", e)
```

```
print("Verejny kluc pre RSA algoritmus bude (n, e) = (", n, ",", e, ")")
```

```
Verejny exponent e =  5
Verejny kluc pre RSA algoritmus bude (n, e) = ( 31546157 , 5 )
```

[22]:
```
# privatny exponent d vypocitam tak, aby platilo
# e * d mod fi(n) == 1
#e = 65537

# na to budem potrebovat rozsireny Euklidov algoritmus
def egcd(a, b):
    u0, u1, v0, v1 = 1, 0, 0, 1
    while b != 0:
        q, a, b = a // b, b, a % b
        u0, u1 = u1, u0 - q * u1
        v0, v1 = v1, v0 - q * v1
    return  a, u0, v0

# Inverzny prvok modulo sa potom vypocita lahko
def modInverse(a, n):
    gcd, u, v = egcd(a, n)
    if gcd == 1:
        return u % n

d = modInverse(e, fi_n)
print("Privatny exponent d = e^(-1) mod fi(n) =", d)
print("e * d mod fi(n) =", e, "*", d, "mod", fi_n, "=", e * d % fi_n)
print("Privatny kluc RSA algoritmu (n, d) = (", n, ",", d, ")")
```

```
Privatny exponent d = e^(-1) mod fi(n) = 25226957
e * d mod fi(n) = 5 * 25226957 mod 31533696 = 1
Privatny kluc RSA algoritmu (n, d) = ( 31546157 , 25226957 )
```

[23]:
```
# Ak chcem niekomu poslat zasifrovanu spravu (cislo)
# potrebujem jeho verejny kluc (prijimatela sifrovanej spravy)

x = 123
y = pow(x, e, n)
print("Spravu x =", x, "zasifrujem pomocou verejneho kluca na y = x^e mod n =",␣
 ↪x, "^", e, "mod", n, "=", y)
```

```
Spravu x = 123 zasifrujem pomocou verejneho kluca na y = x^e mod n = 123 ^ 5 mod
31546157 = 13884799
```

[24]:
```
# Prijimatel spravu desifruje pomocou svojho privatneho kluca

#y = 357341101854457993054768343508
xx = pow(y, d, n)
```

```
print("Sifrovanu spravu y =", y, "desifrujem pomocou privatneho kluca na x =␣
 ↪y^d mod n =", y, "^", d, "mod", n, "=", xx)
```

```
Sifrovanu spravu y = 13884799 desifrujem pomocou privatneho kluca na x = y^d mod
n = 13884799 ^ 25226957 mod 31546157 = 123
```

[25]:
```
# Prelomenie verejneho kluca znamena "uhadnut" prvocisla p a q, z ktorych bolo␣
 ↪vypocitane n
# Pre male to nie je problem pomocou faktorizacie s postupnym delenim

def factorize(n):
    result = []
    i = 2
    while (n > 1):
        j = 0;
        while (n % i == 0):
            n = n / i
            j = j + 1
        if (j > 0):
            result.append((i,j))
        i = i + 1
    return result

n = 1690428486610429
print("Rozklad modulu n =", n, "na prvocinitele:", factorize(n))
```

```
Rozklad modulu n = 1690428486610429 na prvocinitele: [(35352181, 1), (47816809,
1)]
```

[26]:
```
modInverse(5039,10000)
```

[26]:
```
9359
```

[ ]:
```

```

[27]:
```
n = 16812615098258879
print("Rozklad modulu n =", n, "na prvocinitele:", factorize(n))
```

```
Rozklad modulu n = 16812615098258879 na prvocinitele: [(121785061, 1),
(138051539, 1)]
```

[28]:
```
p = 3391
q = 6329
```

[29]:
```
n = p * q
print(n)
```

```
21461639
```

[30]:
```
fi_n = (p-1)*(q-1)
print(fi_n)
```

```
21451920
```

[31]:
```
e = 65537
d = modInverse(e,fi_n)
print(d)
```

```
17131553
```

[32]:
```
y = 6029832903
x = pow(y,d,n)
print(x)
```

```
19826734
```

[ ]: